Each 204 project submission MUST include a write-up with these required sections. In particular, those sections that are highlighted in red:

- Approach, Design, and Algorithm
- Test Cases (Expected vs. Actual)
- JavaFX GUI Test Runs
- JUnit Testing
- Learning Experience

Your project will be imposed with a 20% penalty (of the project grade) unless the write-up includes screen shots of both the Java FX GUI test runs and Junit tests.

Name: Sousanna Chugunova

Instructor: Gary Thai

Project 1

Approach, Design, and Algorithm

I started by writing JUnit test cases for each method I needed to implement. This way, I knew exactly what each method should do before I wrote the code. Once I had my tests in place, I began implementing the methods one by one. For checking uppercase, lowercase, and digits in passwords, I used regular expressions (regex) to make the code as clean and efficient as possible.

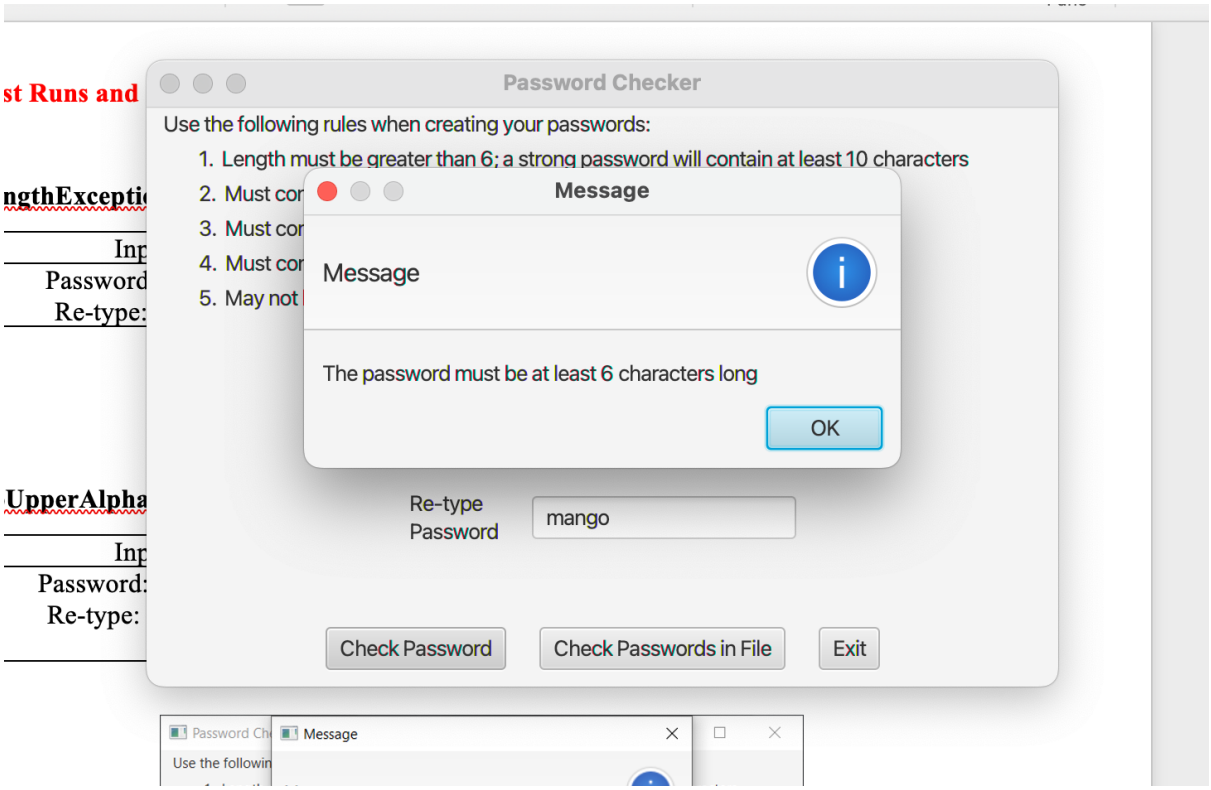
After getting the basic methods working, I created the custom exception classes. These exceptions helped me handle specific errors in a clear and organized way. This step made the isValidPassword method easier to manage, as it could throw the right exceptions in the right order. However, I encountered an issue where the GUI was not appearing. To solve this, I double-checked the GUI-related code, ensured that all necessary resources were properly loaded, and verified that the event dispatch thread was being used correctly. I found that my VM argument was a different version. I found that the version needed was 17.0.1 and I had 15.0.1 Once these issues were resolved, the GUI started working as expected.

Finally, I ran my JUnit tests and made sure everything worked as expected. I had a few syntax errors but after that was resolved, I was left with a running code with a functioning GUI and covering all cases.

Test Runs and Test Cases (Required)

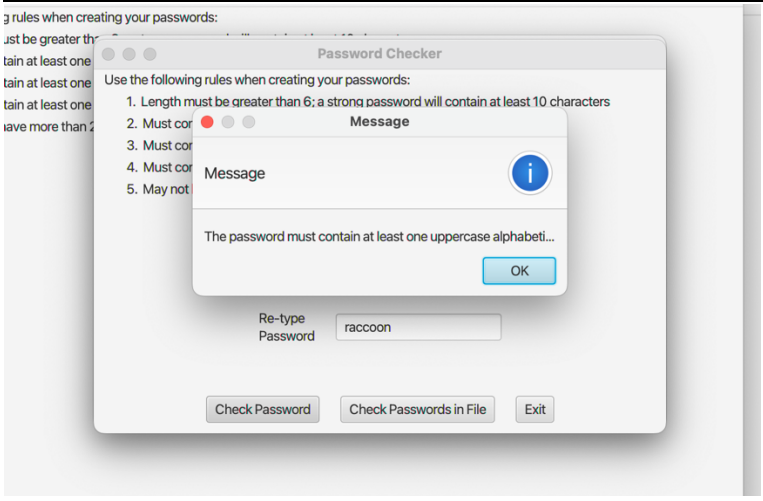
LengthException

Input	Expected Output	Actual Output
Password: mango Re-type: mango	The password must be at least 6 characters long	The password must be at least 6 characters long



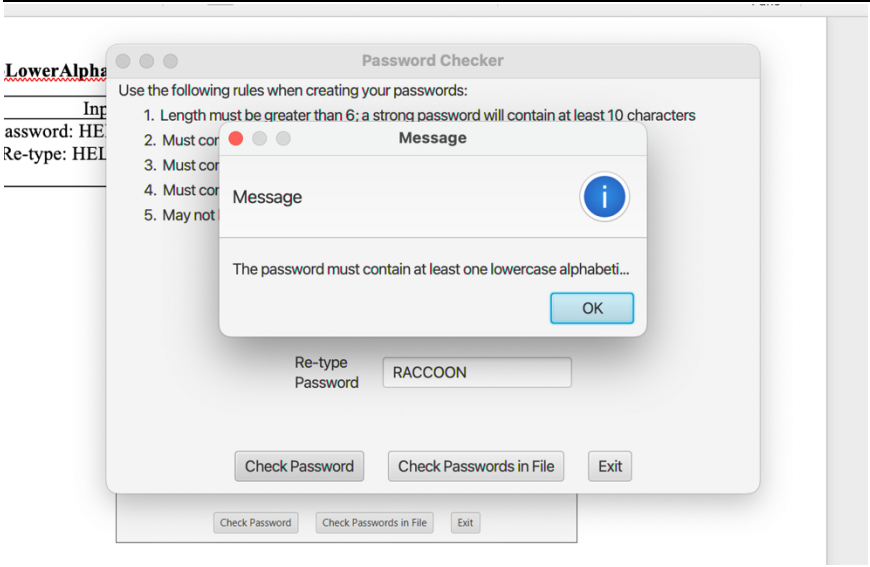
NoUpperAlphaException

Input	Expected Output	Actual Output
Password: raccoon Re-type: raccoon	The password must contain at least one uppercase alphabetic character	The password must contain at least one uppercase alphabetic character



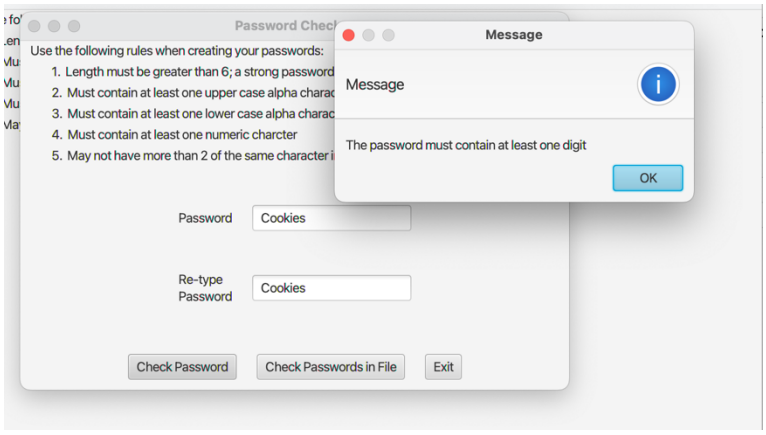
NoLowerAlphaException

Input	Expected Output	Actual Output
Password: RACCOON Re-type: RACCOON	The password must contain at least one lowercase alphabetic character	The password must contain at least one lowercase alphabetic character



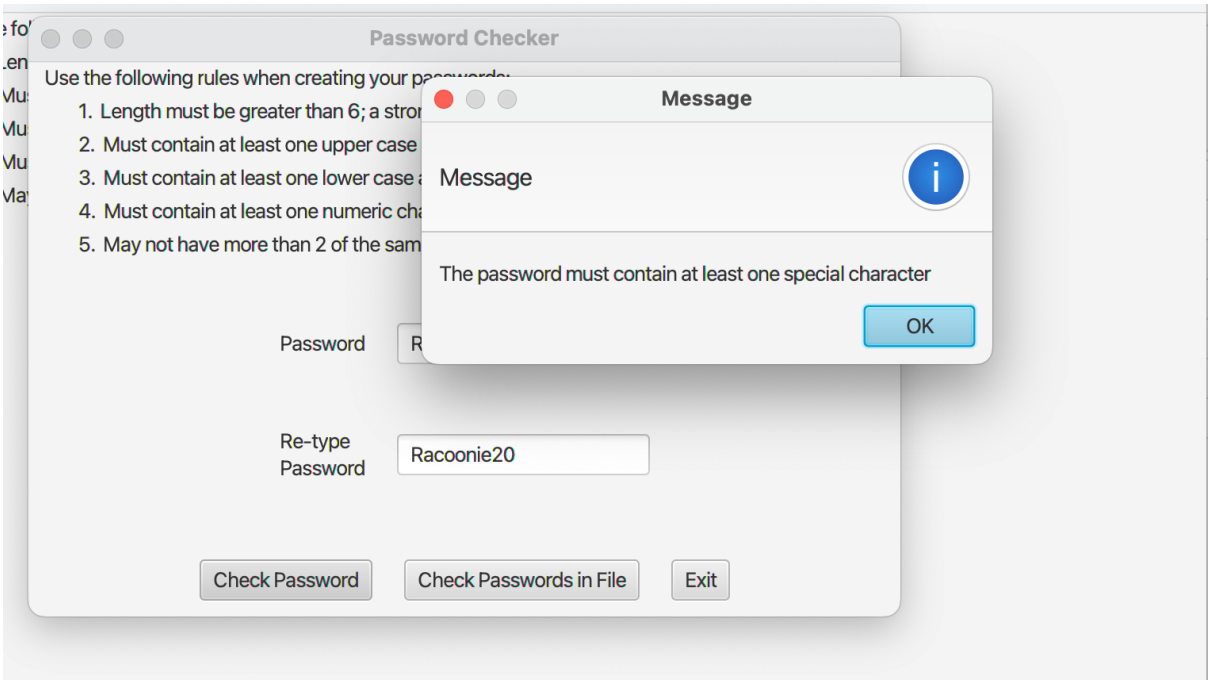
NoDigitException

Input	Expected Output	Actual Output
Password: Cookies Re-type: Cookies	The password must contain at least one digit	The password must contain at least one digit



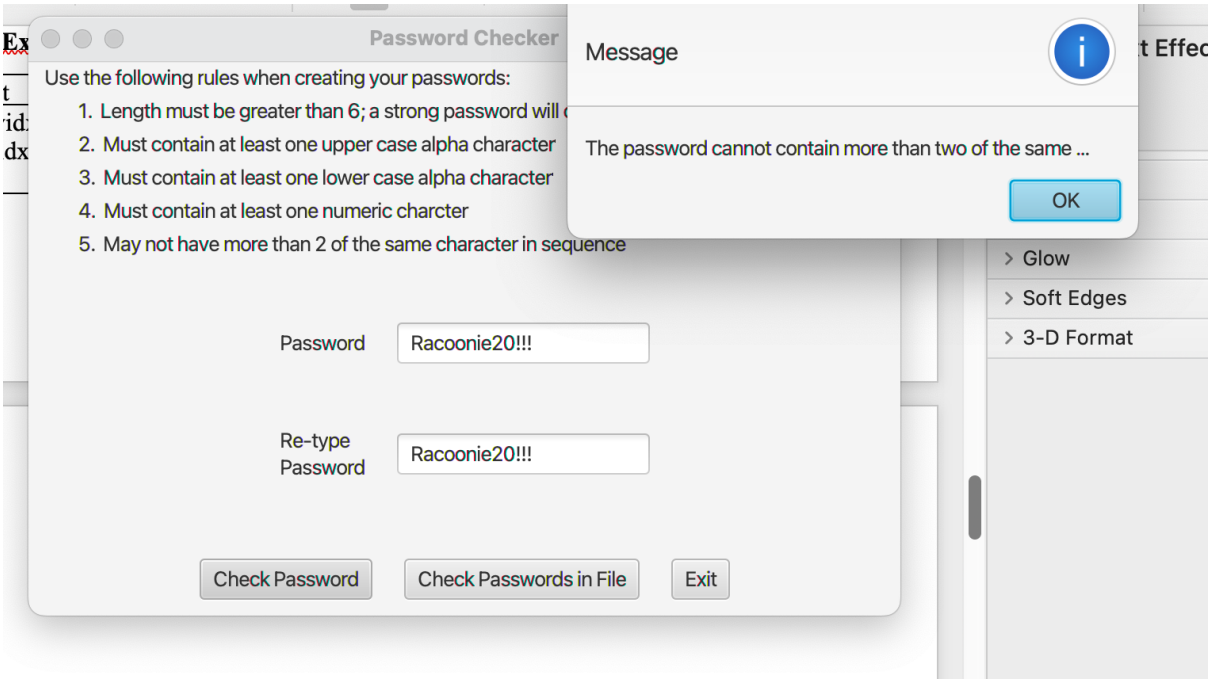
NoSpecialCharacterException

Input	Expected Output	Actual Output
Password: Racoonie20 Re-type: Racoonie20	The password must contain at least one special character	The password must contain at least one special character



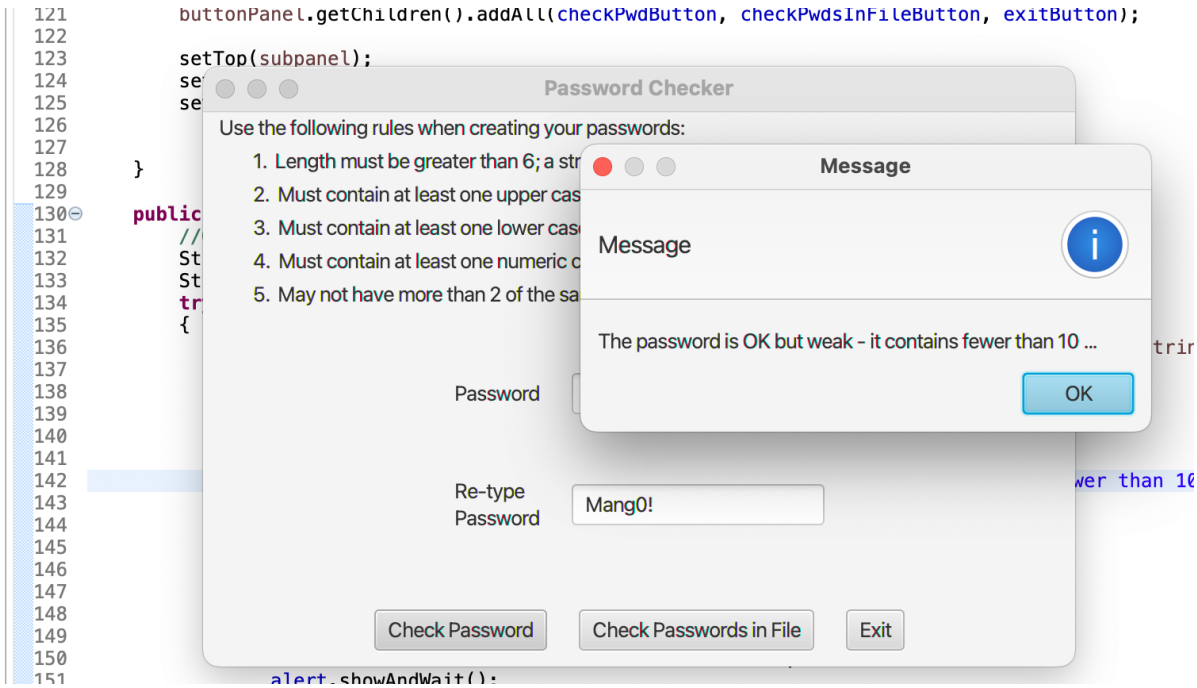
InvalidSequenceException

Input	Expected Output	Actual Output
Password: Racoonie20!!! Re-type: Racoonie20!!!	The password cannot contain more than two of the same character in sequence	The password cannot contain more than two of the same character in sequence



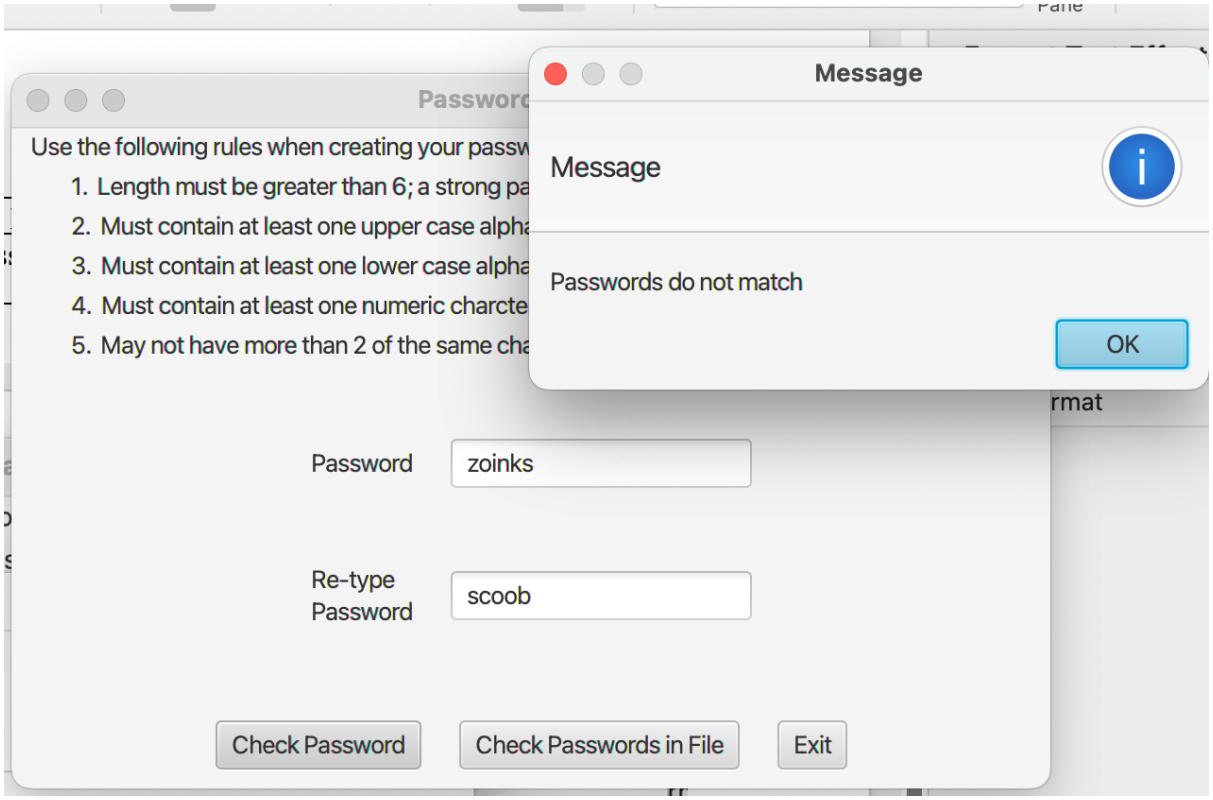
WeakPasswordException

Input	Expected Output	Actual Output
Password: Mang0! Re-4type: Mang0!	The password is OK but weak - it contains fewer than 10 characters	**The password is OK but weak - it contains fewer than 10 characters**



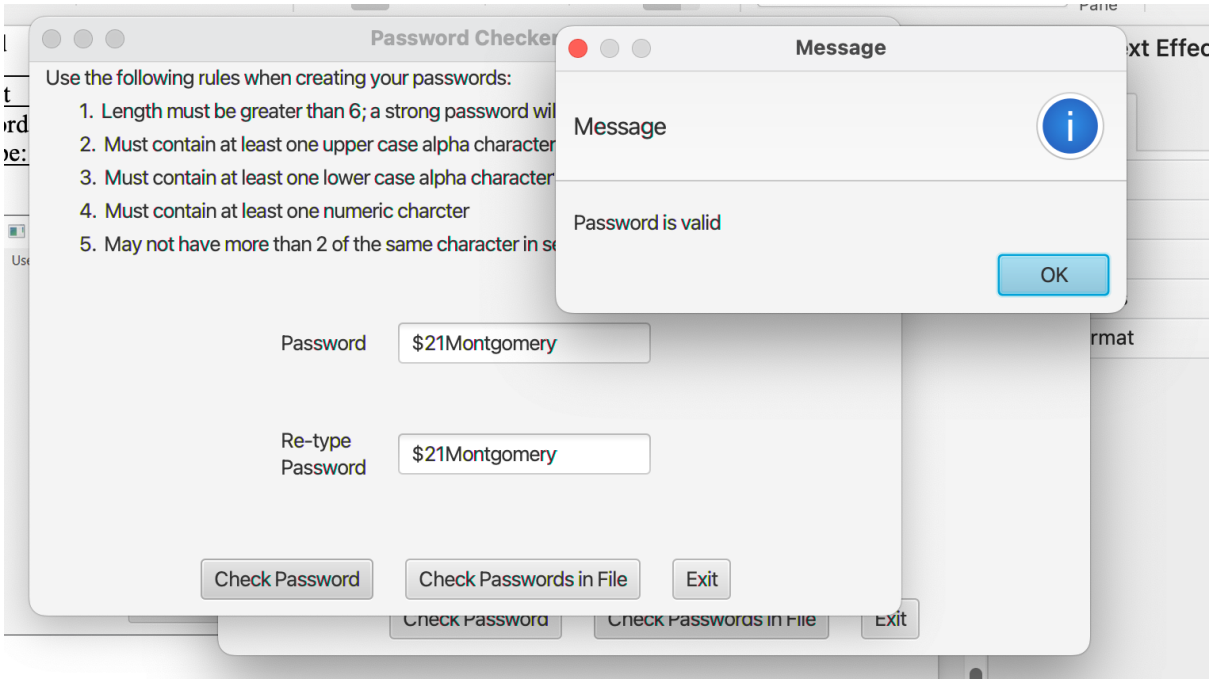
UnmatchedException

Input	Expected Output	Actual Output
Password: zoinks Re-type: scoob	Passwords do not match	**Passwords do not match**



Strong Password

Input	Expected Output	Actual Output
Password: $21Montgomery Re-type: $21Montgomery	Password is valid	Password is valid



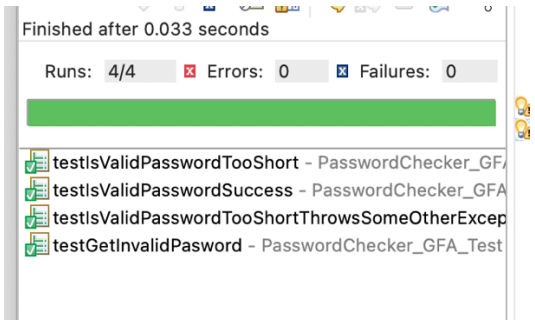
Invalid Passwords From File

(filename: passwords.txt)

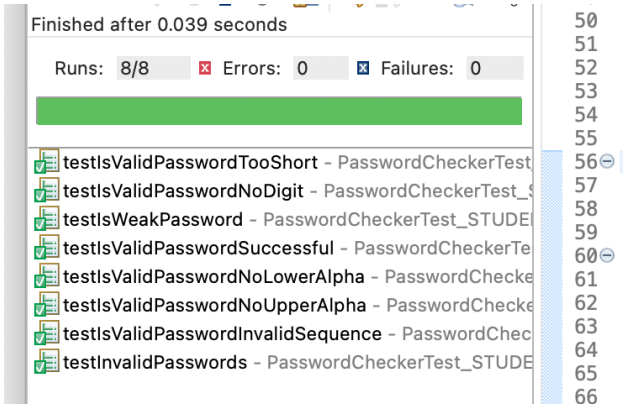
Was not able to get .txt to load on screen.

JUnit Testing (Required)

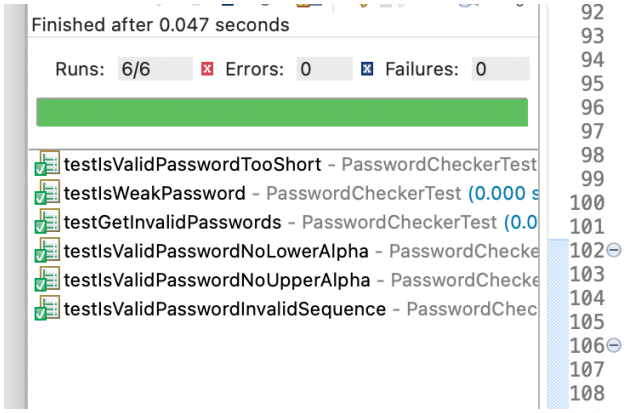
PasswordChecker_GFA_Test.java



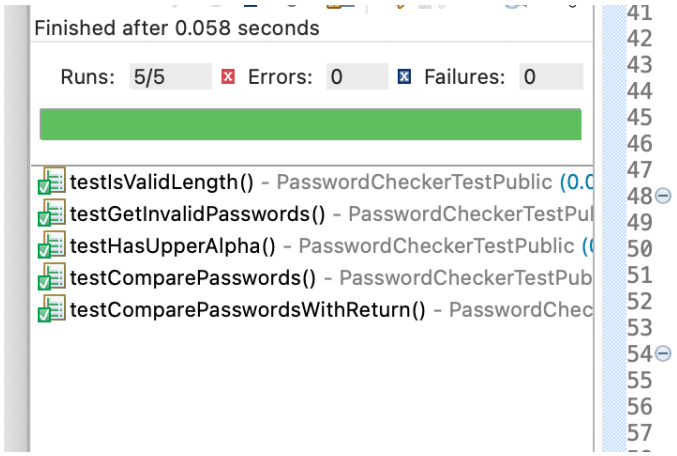
PasswordCheckerTest_STUDENT.java



PasswordCheckerTest.java



PasswordCheckerTestPublic.java



Learning Experience

This project gave me a good review of creating classes and methods using the provided Javadoc files. Most importantly, it got me into the habit of setting up the appropriate coding environment to run JavaFX and JUnit.

Initially, I had trouble getting the GUI to appear, which delayed my progress. I also had issues with my .txt file loading on macOS. This was the first time I used try-catch blocks inside a JUnit file to handle exceptions, which was a new experience. Additionally, creating custom exception classes helped me recall my previous knowledge from the 203 course.

Overall, the project gave me a chance to practice everything I might need for upcoming projects and assignments while also testing my logic. I wouldn't have done anything differently for this project since I gave my best and that I met all the given requirements.

Assumptions

- The user will be using JUnit 5 and have their JavaFX is set up.
- If the user is on MacOs, the .txt file will not appear